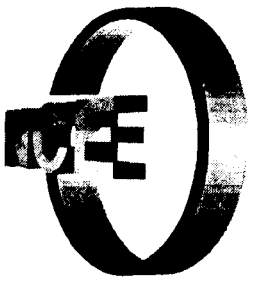

Relatório Técnico

Núcleo de Computação Eletrônica

Logistics Agents

Solução de um Problema Logístico aplicando Sistemas Multi-Agentes

Magali Ribeiro Chaves
Luiz Fernando Rust da Costa Carmo
Luci Pirmez

NCE - 09/99

Universidade Federal do Rio de Janeiro

Logistics Agents

Solução de um Problema Logístico aplicando Sistemas Multi-Agentes

Magali Ribeiro Chaves†, Luci Pirmez e Luiz Fernando Rust da Costa Carmo

NCE/UFRJ – Núcleo de Computação Eletrônica

Universidade Federal do Rio de Janeiro

Tel. : 021 598-3159 – Caixa Postal: 2324 – Rio de Janeiro RJ Brasil

magalichaves@uol.com.br, {luci,rust}@nce.ufrj.br

Resumo

O presente artigo descreve como um conjunto de agentes móveis e cooperativos pode oferecer um melhor desempenho em sistemas que apresentam um contexto distribuído. O objetivo do trabalho é solucionar o problema de distribuição de peças de automóveis, onde uma montadora, com subsidiárias e fornecedores situados em posições geográficas diversas, busca determinar os fornecedores adequados para atender a demanda de suas subsidiárias via Internet. O artigo propõe uma arquitetura composta por agentes que são implementados em um ambiente conhecido como Aglets Workbench .

Este trabalho contribui com um modelo logístico que destaca a importância do uso de agentes móveis, como tecnologia inovadora capaz de mudar o paradigma da programação distribuída.

Abstract

This paper describes how a set of cooperative and mobile agents provide a better performance in distributed context systems. This work proposes the creation of a prototype designed to help the automobile components distribution problem. A car assembler looks for suitable supplies to attend the subsidiaries' orders and this commercial transaction is realized through the Internet. The architecture proposed is composed by agents that were implemented by IBM's Aglets Workbench. This paper contributes with a logistics model which emphasizes the mobile agents technology as a significant and revolutionary paradigm for distributed problem solving.

1. Introdução

O diferencial competitivo de muitas empresas não está simplesmente na qualidade de seus produtos e preços. A distinção também se dá por intermédio da adoção de processos logísticos que respeitem cronogramas pré-estabelecidos, garantindo assim a chegada do produto no tempo previsto. Atentas a exigências como esta, muitas empresas têm feito grandes investimentos em logística.

---

† Aluna do curso de Mestrado em Informática do Instituto de Matemática da Universidade Federal do Rio de Janeiro e funcionária da Universidade do Estado do Rio de Janeiro, lotada no Departamento de Documentação Médica do Hospital Universitário Pedro Ernesto.

A logística, segundo Ballou [2], estuda como prover melhores níveis de rentabilidade nos serviços de distribuição aos clientes e consumidores. Isto é demonstrado através de planejamento, organização e controle efetivo para as atividades de movimentação e armazenagem, visando facilitar o fluxo de produtos. Neste contexto, a logística procura minimizar a separação existente entre oferta e demanda, atendendo os clientes da melhor maneira possível.

No que tange à Ciência da Computação, esta tarefa não é de fácil resolução, devido à quantidade de variáveis e restrições envolvidas no processo. É necessário, portanto, fazer uso de técnicas da Inteligência Artificial (IA).

A Inteligência Artificial Distribuída (IAD) surgiu como uma proposta revolucionária bastante promissora. Ela preocupa-se com a cooperação, coordenação e interação na IA, evoluindo da visão individual para a coletiva, sendo que uma das principais áreas são os Sistemas Multiagentes (SMA) [7], [8].

Nos SMA, os agentes, programas com grau de autonomia, destinados a assistirem ao usuário em determinadas tarefas, interagem entre si, buscando uma solução global.

Como pode ser observado na própria definição de IAD, a interação entre agentes é um dos aspectos mais importantes na resolução de um problema. Contudo, esta forma de comunicação não é de fácil implementação em um sistema real. Tal declaração pode ser comprovada no projeto PIBIC [10], onde este sistema de distribuição de material teve como sub-resultado a constatação deste problema. O protótipo apresentado neste artigo foi criado no ambiente Aglet Workbench que trata a interação entre agentes com bastante transparência, no ponto de vista do programador.

O presente trabalho propõe o desenvolvimento de um sistema, visando solucionar o problema de distribuição de peças de automóveis. Uma montadora, com subsidiárias e fornecedores situados em posições geográficas diversas, tem como missão determinar os fornecedores adequados para atender a demanda de suas subsidiárias, com o objetivo de reduzir o custo total das compras realizadas. A transação comercial é feita na Internet, onde teremos agentes representando tanto os interesses das subsidiárias quanto os dos fornecedores. O destaque maior se dá na implementação de agentes móveis, responsáveis pela melhora de performance exigida no contexto, pois o deslocamento de um objeto móvel em substituição à comunicação remota entre duas entidades, reduz o tráfego na rede.

O trabalho está estruturado da seguinte maneira: na seção 2, encontram-se os conceitos básicos sobre agentes e suas propriedades. Uma breve introdução sobre os diferentes

ambientes de construção de agentes é feita na seção 3, dando maior ênfase ao ambiente Aglets Workbench. Na seção 4, será discutida a arquitetura do sistema e finalmente, na seção 5, são apresentadas as considerações finais.

2. Agentes

O termo "agente" não possui uma definição unificada e universal e, por esta razão, são encontradas na literatura uma série de definições :

- Segundo River [15], um agente "é um programa de computador que funciona em background, e desenvolve tarefas autônomas conforme delegadas pelo usuário" .
- Heilmann [11] descreve o termo agente como "alguém ou alguma coisa que atua como um representante para outro partido, com o propósito expresso de desempenhar ações que são benéficas para a parte representada. É diferenciado de outras aplicações por suas dimensões e capacidade de interagir independentemente da presença do usuário".
- A definição mais adequada à proposta deste projeto é a de Michael Coen[4], onde refere-se a agentes como "programas que travam diálogos, negociam e coordenam transferência de informações".

As definições apresentadas acima tentam descrever o que cada autor entende por agente, baseado nas características que os seus possuem. Um agente deve apresentar algumas propriedades e estas definem qual é sua classificação. A seguir são expostas as propriedades mais relevantes :

- **Autonomia**

  Capacidade de perseguir uma agenda independentemente da presença do usuário. Isso requer ação periódica, execução espontânea e iniciativa, em que o agente precisa ser capaz de tomar ações preemptivas (optativas) e independentes que eventualmente beneficiarão o usuário [6].

- **Mobilidade**

  Habilidade de mover-se de uma máquina para outra, preservando seu estado interno [3].

- **Sociabilidade**

  Capacidade de interagir com outros agentes (e possivelmente humanos) através de algum tipo de linguagem para comunicação de agente [19].

- **Cooperação**

  Capacidade de ter um "espírito" cooperativo. Os agentes trabalham juntos para que possam executar tarefas mutuamente benéficas mais complexas [11].

- **Inteligência**

  Habilidade de negociar efetivamente com ambigüidades[9]. Durante o processo de determinação da ação mais adequada à situação, o agente defronta-se com ambigüidades nos mais diversos níveis.

Neste projeto, os agentes apresentam somente as características de mobilidade, sociabilidade e cooperação.

2.1 Agentes Móveis

Um tipo específico de agentes, denominado comumente como agentes móveis, agentes transportáveis [12] ou agentes itinerantes [5] tem sido objeto de várias pesquisas e é sobre esta classe de agentes que trata este trabalho.

Os agentes móveis são definidos como objetos, possuindo comportamento, estados e localização. Agentes móveis na Internet são programas que podem ser despachados de um computador e transportados para outro, onde irão executar suas tarefas. Chegando na máquina hospedeira, eles apresentam suas credenciais e obtêm acesso a serviços e dados locais. O computador remoto também pode servir como um corretor, reunindo os agentes com interesses semelhantes e metas compatíveis, provendo assim, um lugar no qual os agentes poderão interagir [14].

As vantagens advindas pelo paradigma de agentes móveis são :

- Eficiência – o uso de códigos móveis consome menos recurso de rede devido a possibilidade de migrar para onde existam melhores recursos (cpu, memória, etc);
- Redução de tráfego na rede – a maioria dos protocolos de comunicações envolvem várias interações, especialmente quando medidas de segurança estão habilitadas, com a aplicação de agentes móveis estas interações ocorrem localmente, reduzindo o tráfego na rede;
- Autonomia assíncrona – tarefas podem ser dadas a agentes e estes operam assíncrona e independentemente do programa que os enviou;

- Controle de atividades em tempo real – como os agentes executam localmente, evita-se o problema de latência, intolerável no controle de aplicações de tempo real;
- Robustez e tolerância a falhas – a habilidade dos agentes de reagirem dinamicamente em situações adversas os torna mais tolerantes a falhas;
- Interoperabilidade - desenvolvidos de modo a suportar ambientes heterogêneos, os agentes promove uma maior interoperabilidade;
- Paradigma de desenvolvimento conveniente de um sistema distribuído – o projeto e implementação de sistemas distribuídos pode se tornar mais fácil com o uso de agentes móveis pois estes são inerentemente distribuídos.

Estas vantagens fortalecem as razões deste paradigma estar sendo tão pesquisado e discutido recentemente. Embora não haja um problema específico que somente agentes solucionariam, há que se reconhecer que esta tecnologia trará muitos benefícios em sistemas que apresentam um contexto distribuído.

3. Ambiente de Desenvolvimento dos Agentes

Há uma diversidade de pacotes disponíveis no mercado que permitem aos desenvolvedores construir e gerenciar seus próprios agentes. São eles :

- **Telescript, General Magic**

  O pacote Tabriz Agentware e sua linguagem Telescript são os primeiros projetos sobre ambiente e linguagem de implementação de agentes móveis. Este sistema já foi usado para desenvolvimento de vários produtos comerciais como, por exemplo, PDA da Sony e Envoy da Motorola. É o ambiente mais antigo que está disponível no mercado, tornando-o amadurecido e confiável. Uma de suas desvantagens está na linguagem Telescript não ser plenamente suportada em diferentes plataformas.

- **Agente Tcl, Dartmouth**

  É um sistema de agente móvel baseado na linguagem Tcl e desenvolvido na Universidade de Dartmouth. Ele apresenta uma estrutura bastante nítida; seu código, assim como a documentação e características técnicas estão disponíveis ao público. Agente Tcl implementa alguns mecanismos bastante interessantes sobre segurança. Apesar da linguagem Tcl ser bem suportada em diferentes plataformas, ela é script, tornando-se difícil criar sistemas de agentes móveis muito complexos.

- **Aglet Workbench, IBM**

Os *aglets* (Agente *Applet*) surgiram através de uma pesquisa do laboratório da IBM de Tóquio utilizando a tecnologia de agentes móveis.

Um *aglet* é um objeto em Java que pode mover-se de um *host* para outro. Quando o *aglet* se move, leva seu código de programa bem como seu estado (dados) [1]. Conforme definição de Sommers[17], um *aglet* é considerado um agente móvel pois é um objeto que possui comportamento, estado e localização.

3.1 Ambiente de implementação do projeto – Aglets Workbench

Apesar da existência de vários ambientes de construção de agentes móveis, Aglets Workbench foi escolhido para implementação deste projeto por várias razões :

- Utiliza a linguagem de programação JAVA;
- Implementa a capacidade de mobilidade dos agentes entre computadores;
- Apresenta facilidades de segurança : para um agente mover-se para outra máquina, exige-se que a máquina hospedeira rode um servidor de aglet. Desta forma, alguns problemas relacionados à segurança são solucionados;
- Disponibilidade do ambiente na Internet;
- Facilidade de uso;
- Suporta comunicação assíncrona e síncrona entre agentes locais e/ou remotos.

A composição de um aglet é representada por duas partes distintas : "aglet core" e "aglet proxy", conforme ilustrado na Figura 1. No aglet core estão contidas todas as variáveis e métodos internos de um aglet. Ele oferece interfaces através das quais o aglet pode comunicar-se com seu ambiente. O "Aglet core" é encapsulado por uma "aglet proxy" que age como um escudo, protegendo o aglet contra qualquer tentativa de acesso direto as variáveis e métodos.

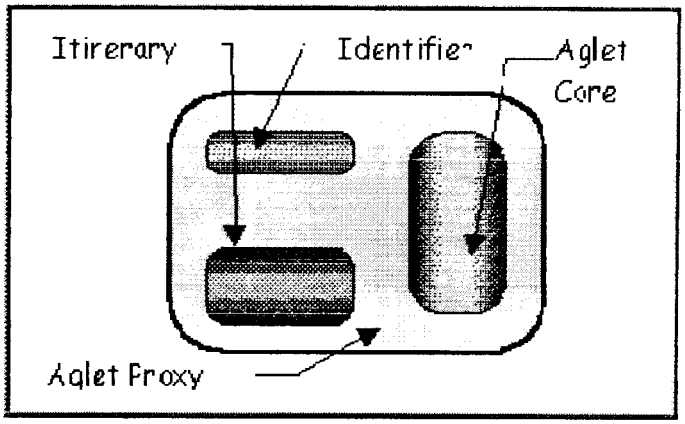

Figura 1 Estrutura de um Aglet

Um *Aglet* pode apresentar vários estados durante seu ciclo de vida. Cada um dos estados corresponde a um método disponível na classe *Aglet*. Esses métodos podem ser sobrescritos , para se implementar o comportamento desejado. São eles:

- ***creation***: inicia um aglet;
- ***cloning***: cria um clone do aglet, com o mesmo estado do original;
- ***dispatching***: transfere um aglet para outra máquina;
- ***retracting***: recupera um *aglet*, previamente despachado;
- ***deactivating***: põe um *aglet* para dormir e seu estado é armazenado no disco;
- ***activating***: reativa um *aglet*, sendo seu estado restabelecido do disco;
- ***dispose***: destrói um *aglet*.

Deve-se lembrar que durante a execução de cada uma das atividades de um *aglet*, pode acontecer o processo de serialização e transmissão pela rede [18].

3.1.1 Java Aglet API

J-AAPI foi desenvolvido por uma equipe de pesquisa do Laboratório da IBM em Tóquio em resposta a necessidade de uma plataforma uniforme para os agentes móveis percorrerem ambientes heterogêneos como a Internet.

A principal vantagem da J-AAPI é que uma vez escrito um *aglet*, ele executará em toda máquina que tenha instalado o pacote J-AAPI, não havendo necessidade de verificar qual o hardware ou o sistema operacional existente ou qual a natureza da implementação particular do J-AAPI no *host* em que o *aglet* está executando [13].

A Aglet API define a funcionalidade fundamental de agentes móveis. As principais classes e interfaces do Java Aglet API, ilustradas na figura 2, são:

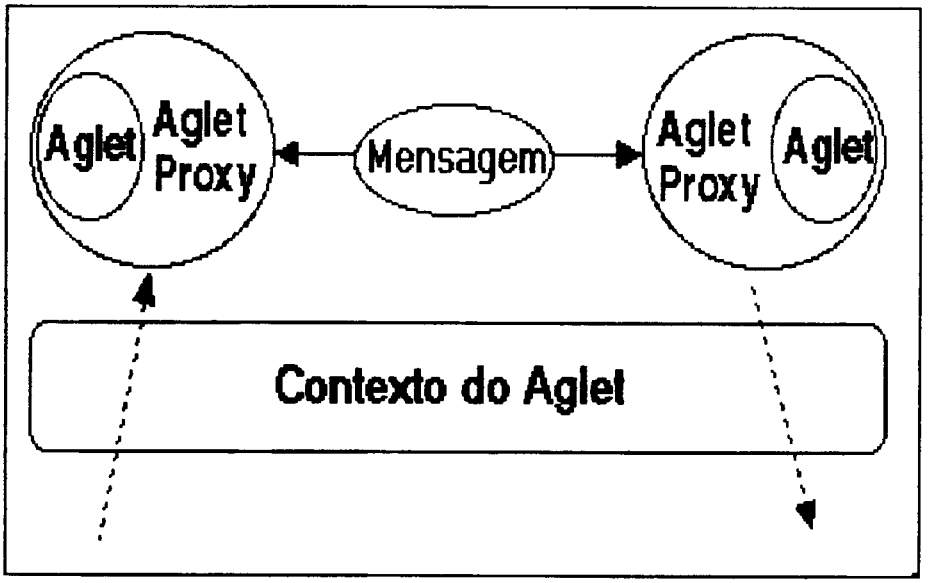

Figura 2 Interface e classes de um Aglet

- aglet.Aglet - A classe abstrata Aglet define os métodos fundamentais (por exemplo, dispatch(URL)) para um agente móvel controlar a sua mobilidade e ciclo de vida. Todos os aglets definidos devem ser instanciados a partir da classe abstrata Aglet. A classe Aglet é usada para se ter acesso aos atributos associados ao *aglet*.
- aglet.AgletProxy - A interface AgletProxy funciona como um manipulador de um *aglet* e provê um modo comum de outros *aglets* terem acesso a um *aglet*. Caso uma classe Aglet tenha vários métodos públicos que não deveriam ser acessados diretamente por outros *aglets*, a maneira para efetuar essa comunicação é através da interface AgletProxy. Para a comunicação com outro *aglet*, deve-se obter inicialmente o objeto *proxy* do primeiro *aglet* e então interagir utilizando esta interface. Assim, o objeto *proxy* do *aglet* age como um objeto de proteção contra outros agentes maliciosos. Quando chamado, a interface do objeto consulta o SecurityManager para determinar se no contexto de execução atual é permitido executar o método, para somente então liberar o acesso. Outro papel importante da interface AgletProxy é proporcionar ao *aglet* transparência de localização.
- aglet.AgletContext - AgletContext provê uma interface para o ambiente de execução que o *aglet* ocupa. Qualquer *aglet* pode obter uma referência para o objeto AgletContext pela primitiva Aglet.getAgletContext() e usar isto para obter informações sobre o local, como o endereço do contexto do *host* e a enumeração de AgletProxies, ou criar um *aglet* novo no contexto.

- aglet.Message - Objetos Aglets comunicam-se trocando objetos da classe Message. Um objeto de mensagem pode ter como argumentos um objeto String para especificar o tipo de mensagem e objetos arbitrários.

4. Sistema de distribuição de peças de automóveis

O presente trabalho propõe a implementação de um protótipo, capaz de solucionar o problema de distribuição de peças, onde uma montadora, com subsidiárias e fornecedores situados em posições geográficas diversas, busca determinar os fornecedores adequados para atender a demanda de suas subsidiárias, com o objetivo de reduzir o custo total das compras realizadas. A relação comercial estabelecida entre subsidiárias e fornecedores é feita via Internet.

Neste sistema, deve-se considerar as seguintes restrições :

a. cada fornecedor, representado pelo setor de autopeças, possui ofertas de produtos com seus custos unitários. Um exemplo é a Metal Leve , fabricante de pistões, bronzinas, buchas, pinos etc.
b. cada subsidiária possui demandas de produtos;
c. O prazo de entrega e a distância entre uma subsidiária e um determinado fornecedor são variáveis a serem consideradas, em virtude do valor do frete influenciar no custo total de compra do produto em oferta.

A tecnologia de agentes é utilizada na criação deste protótipo, onde estes agentes representarão tanto os interesses das subsidiárias, quanto os dos fornecedores. A principal vantagem desta abordagem de especificação é a semelhança do funcionamento do módulo implementado com a realidade.

4.1 Arquitetura do Sistema

A arquitetura do sistema proposto é representada por três tipos de agentes : subsidiária , facilitador e fornecedor .

- **Agente subsidiária**

  Representa os interesses de uma subsidiária. No seu contexto, encontra-se a demanda de peças que uma subsidiária está disposta a adquirir.

Quando o agente subsidiária é criado, abre-se o formulário de Cadastro de Pedido de Material. Neste formulário, o usuário deverá entrar com a identificação da subsidiária, descrição da peça, quantidade a ser requisitada e a informação sobre o grau de urgência da entrega do material, confirmando posteriormente suas informações, conforme ilustrado na Figura 3.

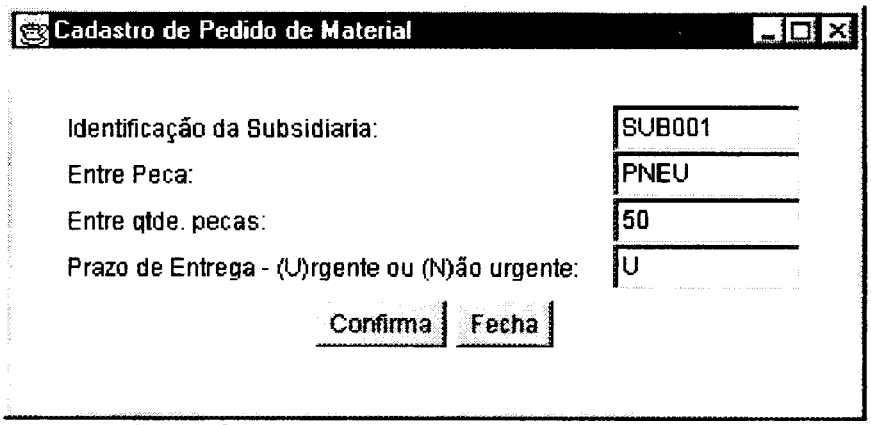

Figura 3 Cadastro de Pedido de Material da Subsidiária

Após a confirmação dos dados digitados no cadastro de pedido de material, o agente subsidiária enviará a mensagem "Register", solicitando seu cadastramento no banco de dados do agente facilitador, conforme apresentado na Figura 4. Quando toda a sua demanda for satisfeita, ela disparará a mensagem "Unregister", solicitando sua saída no cadastro mantido pelo agente facilitador.

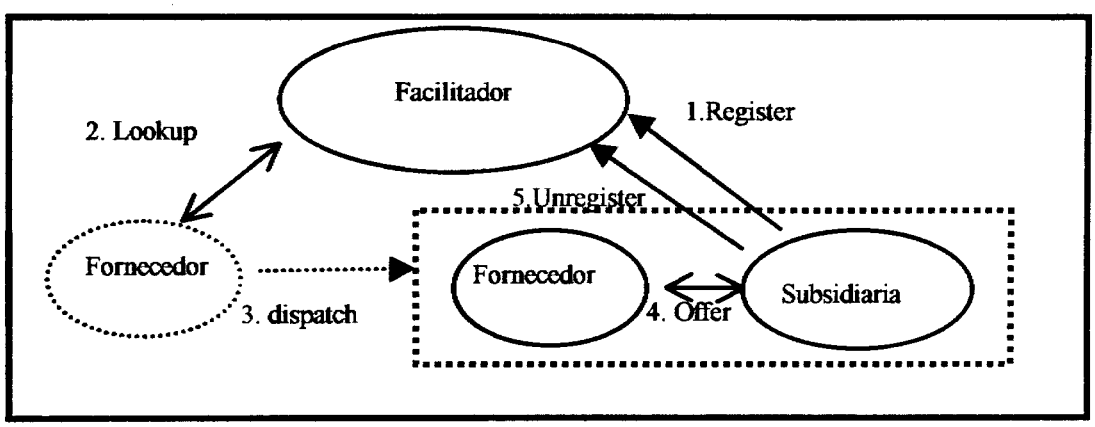

Figura 4 Arquitetura do Sistema

- **Agente facilitador**

  Funciona como intermediador entre agentes subsidiárias e fornecedores. Este agente cadastra todas as subsidiárias com suas respectivas demandas e indica, ao agente fornecedor, a subsidiária que apresenta o menor custo de distribuição para estabelecer o processo de negociação. Para fazer tal indicação, este agente possui um módulo de otimização que procura identificar a subsidiária mais adequada para cada agente fornecedor. Este módulo leva em consideração a distância e o prazo de entrega exigido pela subsidiária; variáveis determinantes que influenciam no custo de distribuição.

- **Agente fornecedor**

  Representa os interesses de um fornecedor. Ele é capaz de mover-se pela rede em busca das subsidiárias indicadas pelo agente facilitador. Quando o agente fornecedor é criado, abre-se o Cadastro de Oferta de Material, ilustrado na Figura 5, onde serão digitados a identificação do fornecedor, descrição da peça oferecida, quantidade de peças disponíveis para comercialização e o valor unitário da peça, informações estas que devem ser confirmadas após sua entrada.

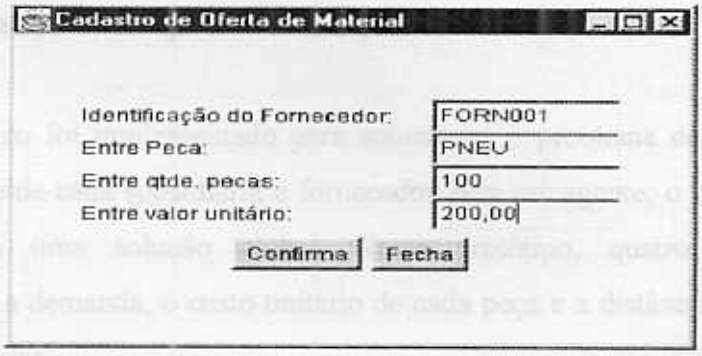

Figura 5 Cadastro de Oferta de Material do agente fornecedor

Após confirmação desses dados, o agente fornecedor entrará em contato com o agente facilitador, através da mensagem "Lookup", e este retornará a subsidiária mais adequada para comprar seus produtos. Obtendo o endereço da subsidiária, o agente fornecedor migrará para máquina desta subsidiária, através do método dispatch e iniciará o processo de negociação, conforme ilustrado na Figura 4.

A Figura 6 apresenta a trajetória do agente fornecedor percorrendo a rede, em busca de subsidiárias.

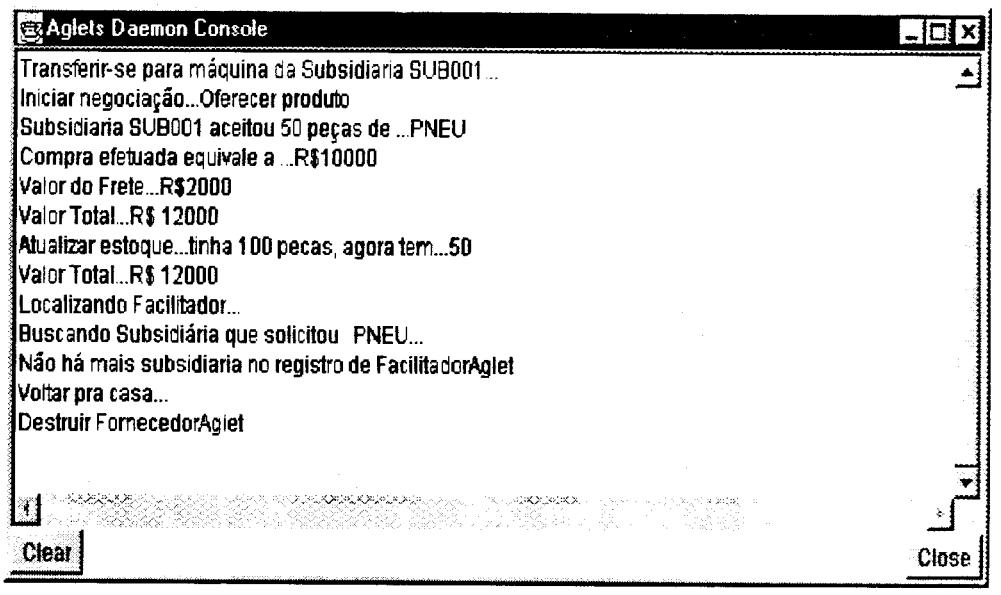

Figura 6 Trajetória do agente fornecedor

Este processo vai se repetir, enquanto o agente fornecedor tiver material ou quando o agente facilitador não possuir mais subsidiárias em seu cadastro. Numa destas situações, o agente fornecedor retornará à sua máquina de origem e será destruído .

5. Considerações finais

Um modelo logístico foi implementado para solucionar o problema de distribuição de peças de automóveis, onde cada subsidiária e fornecedor gera um agente, o qual compartilha informações buscando uma solução global. Neste protótipo, quatro restrições são consideradas : a oferta, a demanda, o custo unitário de cada peça e a distância entre as várias subsidiárias e fornecedores.

A partir da experimentação realizada verificou-se que o sistema proposto apresenta bons resultados, pois a implementação dos agentes móveis evita que a rede fique sobrecarregada.

Na próxima etapa, será desenvolvido um módulo de otimização mais sofisticado, baseado na inteligência computacional, capaz de fornecer uma combinação subsidiária/fornecedor satisfatória, sem a necessidade de testar todo o espaço de busca apresentado no contexto. Segundo pesquisa feita na literatura[16], os Algoritmos Genéticos seriam a técnica ideal para resolver tal tarefa, devido ao seu bom desempenho em problemas de otimização e planejamento.

Referências

[1] Aglet,IBM. Aglets WorkBench. 1997.Endereço Web: http://www.trl.ibm.co.jp/aglets.

[2] Ballou, Ronald H. *"Logística empresarial : transportes, administração de materiais e distribuição física"*. São Paulo : Ed. Atlas, 1993.

[3] Belgrave,Marc. *"The Unified Agent Architecture: A White Paper"*, 1995. http://www.ee.mcgill.ca/~belmarc/uaa_paper.html

[4] Coen, M., Ketchpel, S. , Ramming,C., Kautz, H. e Selman, B. *"An Experiment in the Design of Software Agents"*. AI Principles Research Department AT& T Bell Laboratories. 1997

[5] Chess, D., Grosof, B., Harrison, C., Levine, D., Parris, C. e Tsudik, G. *"Itinerants Agents for Mobile Computing"*. IBM Research Report RC 20010, 1995.

[6] Foner, L. *"What's an Agent, Anyway? A Sociological Case Study"*, Mit Media Lab, 1994. http://foner.www.media.mit.edu/people/foner/Julia.html

[7] Gasser, Les. *"Distribution and coordination of tasks among intelligent agents"*. In: PROCEEDINGS OF THE JCAI'88. Scandinavian Conference on AI. Amsterdam, Springfield, 1988.

[8] Gasser, Les. *"Social conceptions of knowledge and action: DAI foudations and open systems semantics"*, In: ARTIFICIAL INTELLIGENCE - 47. Elsevier Science Publishers: University of Southern California, p. 107-138. Los Angeles. 1990.

[9] Gilbert, D., Aparicio, Manny, A. *"Intelligent Agent Strategic"*. 1996. http://activist.gpl.ibm.com:81/whitePaper/ptc2.html

[10] Gonçalvez, Alexandre Leopoldo. *"Desenvolvimento de um Ambiente de Bases de Conhecimento Distribuídas"*. Blumenau, DSC/FURB, 1997.

[11] Heilmann, K., Kihanya, D., Light, A. e Musembya, P. *"Intelligent Agents: A Technology and Business Aplication Analysis"*. 1995. http://haas.berkeley.edu/~heilmann/agents/index.html

[12] Kotay, K. e Kotz,D. *"Transportable Agents"*, Department of Computer Science – Dartmouth College, 1994.

[13] Lange, Danny B. e Oshima, Mitsuru. *"Programming Mobile Agents in JavaTM - with the Java Aglet API"*. 1997. http://www.trl.ibm.co.jp/aglets.

[14] Oshima, M.; Karjoth, G. Aglets Specification (Alpha 5) Draft. 1997. http://www.trl.ibm.com/aglets/spec_alpha5.html

[15] River. "*Agent Technology*". 1996. http://www.opensesame.com/webpages/sesame/whitepaper.html

[16] Syswerda, G. "Schedule optimization using Genetic Algorithms", em "Handbook of Genetic Algorithms", editado por Lawrence Davis, Van Nostrand Reinhold, New York, 1991.

[17] Sommers. *"Breat Agents : not just for Bond anymore"*. 1997. http://www.javaworld.com/javaworld/jw-04-1197/jw-04-agents.html.

[18] Venners,Bill. *"Under the Hood: The architecture of aglets"*. 1997. http://www.javaworld.com/javaworld/jw-04-1997/jw-04-hood.html.

[19] Wooldridge, M.J., e Jeennings, N. R. "*Intelligent Agents: Theory and Practice*", Knowledge Engineering Review,1994. http://www.doc.mmu.ac.uk/STAFF/mike/ker95.ps